What is Claimed:

1. A computer-implemented method of compiling computer code, comprising:

creating a flowgraph according to abstract computer instructions, wherein the flowgraph has a plurality of basic blocks and at least one data object, and wherein the abstract instructions are translated from a parse tree formed from computer code;

assigning a depth-first order to the plurality of basic blocks;

determining a dominance relationship between the plurality of basic blocks;

determining whether any loops are present within the flowgraph and, if any loops are present, identifying the loops;

determining a usage of the at least one data object;

determining a creation point, destruction point and lock point for the at least one data object according to the usage, identified loops, dominance relationship and depth-first order of the plurality of basic blocks; and

inserting instructions into the computer code to create the at least one data object at the creation point, to destroy the at least one data object at the destruction point and to lock the at least one data object at the lock point.

2. The computer-implemented method of claim 1, wherein determining a creation point further comprises:

identifying a first use of the at least one data object and identifying a first basic block in which the first use occurs from the plurality of basic blocks, wherein the first use of the at least one data object is the original creation point;

identifying a use of the at least one data object subsequent to the first use and identifying a second basic block in which the subsequent use occurs from the plurality of basic blocks;

calculating an intersection of the pre-dominators of the first basic block with the pre-dominators of the second basic block;

determining whether the intersection contains the first basic block; and

choosing, if the intersection does not contain the first basic block, a new creation point from the intersection.

3. The computer-implemented method of claim 2, wherein choosing a new creation point from the intersection comprises determining whether the new creation point is in a different loop

than the original creation point and, if so, choosing a new creation point that bypasses all inner loops between the original creation point and the new creation point.

4. The computer-implemented method of claim 1, wherein determining a destruction point comprises:

identifying a last use of the at least one data object and identifying a first basic block in which the last use occurs from the plurality of basic blocks, wherein the last use of the at least one data object is the original destruction point;

identifying a use of the at least one data object previous to the last use and identifying a second basic block in which the previous use occurs from the plurality of basic blocks;

calculating an intersection of the post-dominators of the first basic block with the post-dominators of the second basic block;

determining whether the intersection contains the first basic block; and

choosing, if the intersection does not contain the first basic block, a new destruction point from the intersection.

5. The computer-implemented method of claim 4, wherein choosing a new destruction point from the intersection comprises determining whether the new destruction point is in a different loop than the original destruction point and, if so, choosing a new destruction point that bypasses all inner loops between the original destruction point and the new destruction point.

6. The computer-implemented method of claim 1, wherein determining a lock point comprises:

determining a set of at least one data object;

creating a first set of basic blocks from the plurality of basic blocks that write to the at least one data object;

creating a second set of basic blocks from the plurality of basic blocks that read from the at least one data object;

removing from the first and second sets any of the plurality of basic blocks that are not contained within a synchronized scope; and

determining a type of lock to place for each of the at least one data object.

7. The computer-implemented method of claim 6, wherein determining a type of lock for each of the at least one data object comprises selecting a read lock for the at least one data object if the first set is empty, otherwise, selecting a write lock.

8. The computer-implemented method of claim 6, wherein determining a lock point further comprises:

identifying a first use of the at least one data object and identifying a first basic block in which the first use occurs from the plurality of basic blocks, wherein the first use of the at least one data object is the original lock point;

identifying a use of the at least one data object subsequent to the first use and identifying a second basic block in which the subsequent use occurs from the plurality of basic blocks;

calculating an intersection of the pre-dominators of the first basic block with the pre-dominators of the second basic block;

determining whether the intersection contains the first basic block; and

choosing, if the intersection does not contain the first basic block, a new lock point from the intersection.

9. The computer-implemented method of claim 8, wherein choosing a new lock point from the intersection comprises determining whether the new lock point is in a different loop than the original lock point and, if so, choosing a new lock point that bypasses all inner loops between the original lock point and the new lock point.

10. The computer-implemented method of claim 1, wherein the computer code is XLANG/s.

11. The computer-implemented method of claim 1, wherein the at least one data object is a variable.

12. The computer-implemented method of claim 1, wherein the at least one data object is a symbol.

13. The computer-implemented method of claim 1, wherein the at least one data object is a message.

14. The computer-implemented method of claim 1, wherein the flowgraph corresponds to a long-running transaction.

15. The computer-implemented method of claim 1, wherein the flowgraph corresponds to an atomic transaction.

16. The computer-implemented method of claim 1, wherein the flowgraph corresponds to a long-running transaction with an exception handler.

17. The computer-implemented method of claim 1, wherein the flowgraph corresponds to a long-running transaction with compensation.

18. The computer-implemented method of claim 1, wherein the flowgraph corresponds to an atomic transaction with compensation.

19. The computer-implemented method of claim 1, wherein the flowgraph corresponds to a long-running transaction with an exception handler and compensation.

20. A computer-readable medium having computer-executable instructions for compiling computer code, the method comprising:

creating a flowgraph according to abstract computer instructions, wherein the flowgraph has a plurality of basic blocks and at least one data object, and wherein the abstract instructions are translated from a parse tree formed from computer code;

assigning a depth-first order to the plurality of basic blocks;

determining a dominance relationship between the plurality of basic blocks;

determining whether any loops are present within the flowgraph and, if any loops are present, identifying the loops;

determining a usage of the at least one data object;

determining a creation point, destruction point and lock point for the at least one data object according to the usage, identified loops, dominance relationship and depth-first order of the plurality of basic blocks; and

inserting instructions into the computer code to create the at least one data object at the creation point, to destroy the at least one data object at the destruction point and to lock the at least one data object at the lock point.

21. The computer-readable medium of claim 20, wherein determining a creation point further comprises:

identifying a first use of the at least one data object and identifying a first basic block in which the first use occurs from the plurality of basic blocks, wherein the first use of the at least one data object is the original creation point;

identifying a use of the at least one data object subsequent to the first use and identifying a second basic block in which the subsequent use occurs from the plurality of basic blocks;

calculating an intersection of the pre-dominators of the first basic block with the pre-dominators of the second basic block;

determining whether the intersection contains the first basic block; and

choosing, if the intersection does not contain the first basic block, a new creation point from the intersection.

22. The computer-readable medium of claim 21, wherein choosing a new creation point from the intersection comprises determining whether the new creation point is in a different loop than the original creation point and, if so, choosing a new creation point that bypasses all inner loops between the original creation point and the new creation point.

23. The computer-readable medium of claim 20, wherein determining a destruction point comprises:

identifying a last use of the at least one data object and identifying a first basic block in which the last use occurs from the plurality of basic blocks, wherein the last use of the at least one data object is the original destruction point;

identifying a use of the at least one data object previous to the last use and identifying a second basic block in which the previous use occurs from the plurality of basic blocks;

calculating an intersection of the post-dominators of the first basic block with the post-dominators of the second basic block;

determining whether the intersection contains the first basic block; and

choosing, if the intersection does not contain the first basic block, a new destruction point from the intersection.

24. The computer-readable medium of claim 23, wherein choosing a new destruction point from the intersection comprises determining whether the new destruction point is in a different

loop than the original destruction point and, if so, choosing a new destruction point that bypasses all inner loops between the original destruction point and the new destruction point.

25. The computer-readable medium of claim 20, wherein determining a lock point comprises:

determining a set of at least one data object;

creating a first set of basic blocks from the plurality of basic blocks that write to the at least one data object;

creating a second set of basic blocks from the plurality of basic blocks that read from the at least one data object;

removing from the first and second sets any of the plurality of basic blocks that are not contained within a synchronized scope; and

determining a type of lock to place for each of the at least one data object.

26. The computer-readable medium of claim 25, wherein determining a type of lock for each of the at least one data object comprises selecting a read lock for the at least one data object if the first set is empty, otherwise, selecting a write lock.

27. The computer-readable medium of claim 25, wherein determining a lock point further comprises:

identifying a first use of the at least one data object and identifying a first basic block in which the first use occurs from the plurality of basic blocks, wherein the first use of the at least one data object is the original lock point;

identifying a use of the at least one data object subsequent to the first use and identifying a second basic block in which the subsequent use occurs from the plurality of basic blocks;

calculating an intersection of the pre-dominators of the first basic block with the pre-dominators of the second basic block;

determining whether the intersection contains the first basic block; and

choosing, if the intersection does not contain the first basic block, a new lock point from the intersection.

28. The computer-readable medium of claim 27, wherein choosing a new lock point from the intersection comprises determining whether the new lock point is in a different loop than the original lock point and, if so, choosing a new lock point that bypasses all inner loops between the original lock point and the new lock point.

29. The computer-readable medium of claim 20, wherein the computer code is XLANG/s.

30. The computer-readable medium of claim 20, wherein the at least one data object is a variable.

31. The computer-readable medium of claim 20, wherein the at least one data object is a symbol.

32. The computer-readable medium of claim 20, wherein the at least one data object is a message.

33. The computer-readable medium of claim 20, wherein the flowgraph corresponds to a long-running transaction.

34. The computer-readable medium of claim 20, wherein the flowgraph corresponds to an atomic transaction.

35. The computer-readable medium of claim 20, wherein the flowgraph corresponds to a long-running transaction with an exception handler.

36. The computer-readable medium of claim 20, wherein the flowgraph corresponds to a long-running transaction with compensation.

37. The computer-readable medium of claim 20, wherein the flowgraph corresponds to an atomic transaction with compensation.

38. The computer-readable medium of claim 20, wherein the flowgraph corresponds to a long-running transaction with an exception handler and compensation.

39. A method of compiling XLANG/s code, comprising:

creating a flowgraph having a plurality of basic blocks and at least one data object according to abstract computer instructions;

assigning a depth-first order to the plurality of basic blocks;

determining a dominance relationship between the plurality of basic blocks;

identifying any loops formed by the plurality of basic blocks;

determining a usage of the at least one data object according to the abstract instructions;

determining a creation point, destruction point and lock point for the at least one data object according to the usage, identified loops, dominance relationship and depth-first order of the plurality of basic blocks; and

inserting instructions into the computer code to create the at least one data object at the creation point, to destroy the at least one data object at the destruction point and to lock the at least one data object at the lock point.

40. The method of claim 39, wherein determining a creation point further comprises:

identifying a first use of the at least one data object and identifying a first basic block in which the first use occurs, wherein the first use of the at least one data object is the original creation point;

identifying a second use of the at least one data object in a second basic block, wherein the second basic block is a higher-ordered basic block than the first basic block;

calculating an intersection of the pre-dominators of the first basic block with the pre-dominators of the second basic block;

determining whether the intersection contains the first basic block; and

choosing, if the intersection does not contain the first basic block, a new creation point contained in the intersection.

41. The method of claim 40, wherein choosing a new creation point from the intersection comprises determining whether the new creation point is in a different loop than the original creation point and, if so, choosing a new creation point that bypasses all inner loops between the original creation point and the new creation point.

42. The method of claim 39, wherein determining a destruction point comprises:

identifying a last use of the at least one data object and identifying a first basic block in which the last use occurs, wherein the last use of the at least one data object is the original destruction point;

identifying a use of the at least one data in a second basic block, wherein the second basic block is a lower-ordered basic block than the first basic block;

calculating an intersection of the post-dominators of the first basic block with the post-dominators of the second basic block;

determining whether the intersection contains the first basic block; and

choosing, if the intersection does not contain the first basic block, a new destruction point contained in the intersection.

43. The method of claim 42, wherein choosing a new destruction point from the intersection comprises determining whether the new destruction point is in a different loop than the original destruction point and, if so, choosing a new destruction point that bypasses all inner loops between the original destruction point and the new destruction point.

44. The method of claim 39, wherein determining a lock point comprises:

determining a set of at least one data object;

creating a first set of basic blocks that write to the at least one data object;

creating a second set of basic blocks that read from the at least one data object;

removing from the first and second sets any of the plurality of basic blocks that are not contained within a synchronized scope; and

determining a type of lock to place for each of the at least one data object.

45. The method of claim 44, wherein determining a type of lock for each of the at least one data object comprises selecting a read lock for the at least one data object if the first set is empty, otherwise, selecting a write lock.

46. The method of claim 44, wherein determining a lock point further comprises:

identifying a first use of the at least one data object and identifying a first basic block in which the first use occurs, wherein the first use of the at least one data object is the original lock point;

identifying a use of the at least one data object in a second basic block, wherein the second basic block is a higher-ordered basic block than the first basic block;

calculating an intersection of the pre-dominators of the first basic block with the pre-dominators of the second basic block;

determining whether the intersection contains the first basic block; and

choosing, if the intersection does not contain the first basic block, a new lock point contained in the intersection.

47. The method of claim 46, wherein choosing a new lock point from the intersection comprises determining whether the new lock point is in a different loop than the original lock point and, if so, choosing a new lock point that bypasses all inner loops between the original lock point and the new lock point.